AMENDMENTS TO THE CLAIMS

1. (Currently Amended) A processor comprising

memory;

decode logic to receive a launch instruction; and

one or more execution units to execute the launch instruction by loading ~~load~~ an authenticated code module into the memory, ~~to lock~~ locking the memory, ~~to retrieve~~ retrieving a key, ~~to authenticate~~ authenticating the authenticated code module stored in the memory using the key, and ~~to execute~~ initiate execution of the authenticated code module stored in the memory ~~in response to executing a launch instruction~~.

2. (Previously Presented) The processor of claim 1 further comprising a cache memory that provides the memory.

3. (Cancelled)

4. (Previously Presented) The processor of claim 2 wherein the execution units lock the cache memory to prevent replacement of lines of the authenticated code module stored in the cache memory.

5. (Previously Presented) The processor of claim 1 wherein the execution units lock the memory to prevent other processors from altering the authenticated code module stored in the memory.

6. (Currently Amended) The processor of claim 1 ~~further comprising a decoder~~ wherein the decode logic is also to generate one or more opcodes for the launch instruction, wherein the execution units authenticate and execute the authenticated code module in response to executing the one or more opcodes.

7. (Cancelled)

8. (Previously Presented) The processor of claim 1, wherein the execution units retrieve the key specified by one or more operands of the launch instruction.

9. (Previously Presented) The processor of claim 1, wherein the execution units, in response to the launch instruction, retrieve the key from a chipset.

10. (Withdrawn) The processor of claim 1, wherein the execution units, in response to the launch instruction, retrieve a key from a token and use the key to authenticate the authenticated code module stored in the protected memory.

11. (Withdrawn) The processor of claim 1, wherein the execution units, in response to the launch instruction, use a key of the processor to authenticate the authenticated code module stored in the protected memory.

12. (Previously Presented) The processor of claim 1, wherein the execution units, in response to the launch instruction, decrypt at least a portion of the authentication module stored in the memory.

13. (Original) The processor of claim 1, wherein the execution units, in response to the launch instruction, decrypt at least a portion of the authentication module to obtain a digest value, and determine whether the authentication module is authentic based upon the digest value.

14. (Original) The processor of claim 1, wherein the execution units, in response to the launch instruction, obtain a digest value for the authentication code module, generate a computed digest value from at least a portion of the authenticated code module, and determine that the authenticated code module is authentic in response to the digest value and the computed digest value having a predetermined relationship.

15. (Original) The processor of claim 1, wherein the execution units, in response to the launch instruction, RSA-decrypt a signature of the authentication code module to obtain a digest value from the signature, perform a SHA-1 hash on the authenticated code module to generate a computed digest value, and determine that the authenticated code module is authentic in response to the digest value and the computed digest value being equal.

16. (Original) The processor of claim 1, wherein the execution units initiate execution of the authenticated code module only if the authenticated code module is determined to be authentic.

17. (Original) The processor of claim 16, wherein the execution units generate an error code in response to determining that the authenticated code module is not authentic.

18. (Original) The processor of claim 17, wherein the execution units generate a trap in response to determining that the authenticated code module is not authentic.

19. (Withdrawn) The processor of claim 1, wherein the execution units execute the authenticated code module from a execution point specified by one or more operands of the launch instruction.

20. (Withdrawn) The processor of claim 1, wherein the execution units execute the authenticated code module from an execution point specified by one or more fields of the authenticate code module.

21. (Withdrawn) The processor of claim 1, wherein the execution units mask one or more events selected from a group of events comprising INTR, NMI, SMI, INIT, and A20M events in response to executing the launch instruction.

22. (Previously Presented) The processor of claim 1, wherein the execution units authenticate and initiate execution of the authenticated code module stored in the memory in response to executing microcode associated with the launch AC instruction.

23. (Original) The processor of claim 1, embodied in a machine readable medium.

24. (Cancelled)

25. (Withdrawn) The processor of claim 24, wherein the front end generates one or more ops for the instruction, and execution of the instruction results in the execution units executing the one or more ops.

26. (Withdrawn) The processor of claim 24 further comprising a processor key, wherein execution of the instruction results in the execution units authenticating the authenticated code module based upon the processor key.

27. (Cancelled)

28. (Cancelled)

29. (Cancelled)

30. (Withdrawn) The processor of claim 28, wherein execution of the instruction results in the execution units initiating execution of the authenticated code module from an execution point specified by one or more operands of the instruction.

31. (Withdrawn) The processor of claim 24, wherein execution of the instruction results in the execution units initiating execution of the authenticated code module from an execution point specified by one or more fields of the authenticate code module.

32. (Cancelled)

33. (Cancelled)

34. (Cancelled)

35. (Cancelled)

36. (Cancelled)

37. (Cancelled)

38. (Cancelled)

39. (Cancelled)